RUN-TIME RECONFIGURATION METHOD FOR PROGRAMMABLE UNITS

FIELD OF THE INVENTION

The present invention relates to reprogrammable units.

BACKGROUND INFORMATION

Programmable units with a two- or multi-dimensional cell architecture (in particular FPGAs, DPGAs and DFPs, etc.) are programmed today in two different ways.

- Once, i.e., the configuration cannot be changed after programming. All the configured elements of the unit thus perform the same function over the entire period of time during which the application is being carried out.

- During operation, i.e., the configuration can be changed after installation of the unit, by loading a configuration data file, at the start of the application. Units such as, for example, FPGA units, cannot be reconfigured during operation. With reconfigurable units, further processing of data during the reconfiguration is usually impossible, and the required reconfiguration time is much too long.

In addition to FPGAs, there are also DPGAs. These units store a number of different configurations which are selected by a special data packet. Run-time reconfiguration of these memory devices is impossible.

Major problems are posed by run-time reconfiguration of all programmable units or parts thereof, especially synchronization. All the possibilities proposed so far involve stopping the processing of the entire unit during

EL179105277

reconfiguration. Another problem is selection of the new subconfiguration to be loaded and integration of this subconfiguration into the existing configuration.

SUMMARY OF THE INVENTION

The method according to the present invention makes it possible to reload a run-time reconfigurable unit efficiently and without having any effect on the areas not involved in the reconfiguration. In addition, this method makes it possible to select configurations as a function of the prevailing configuration. The problem of synchronizing the areas involved in the reconfiguration with those not involved in the reconfiguration is also solved.

In accordance with an exemplary embodiment of the present invention, a method is provided for reconfiguring programmable units having a two- or multi-dimensional cell arrangement. The method of the present invention makes it possible to reconfigure the unit(s) without limiting the operability of the cells not involved in the reconfiguration. This method makes it possible to load complete configurations or subconfigurations into the programmable unit(s).

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a flow chart of the steps to be carried out after a system start in accordance with an exemplary embodiment of the present invention.

Figure 2 shows a flow chart of the steps to be carried out after a reconfiguration request is received in accordance with the exemplary embodiment of the present invention.

Figure 3 shows a flow chart of the steps to be carried out in the FIFO memory processing in accordance with the exemplary embodiment of the present invention.

Figure 4 shows a flow chart of the steps to be carried

out in configuring the cells in accordance with the exemplary embodiment of the present invention.
Figure 5 shows the PLU with its registers, as well as the configuration memory, the subdivision into jump table, start configuration, additional configurations and the FIFO memories, in accordance with the exemplary embodiment of the present invention.
Figure 6 shows two details from a configuration program and four details from the jump table and how they are related in time in accordance with the exemplary embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

The method in accordance with the exemplary embodiment that is described herein presupposes a programmable unit which has the following properties:

- **Primary logic unit:** The primary logic unit (PLU) is the part of the unit that performs the loading and entering of configuration strings into the elements of the unit to be configured (cells).
- **Cells:** The unit has a number of cells that can be addressed individually by the PLU.
- **Feedback to the PLU:** Each cell or group of cells must be able to notify the PLU whether it can be reconfigured.
- **Feedback to cells:** Each cell must have the option of sending a STOP signal to the cells from which it has received its data to be processed.
- **START/STOP identifier:** Each cell must have the possibility of setting a START/STOP identifier.
  - The START identifier characterizes a cell as the start of a longer processing chain (macro).
  - The STOP identifier marks the end of the macro, i.e., the point at which the processing of the macro has yielded a result.

Structure of a configuration string: The PLU in accordance with the exemplary embodiment of the present invention is a state machine that can process configuration strings.

In addition to configuration strings for cells, there are entries which can be recognized as commands by the PLU. It is thus possible to differentiate whether the contents of the configuration string are to be transmitted to a cell or whether they represent a command for the state machine.

A configuration string which is transmitted to cells of the unit must contain at least the following data:

- Address of the cell, e.g., as linear numbers or as X, Y coordinates.
- Configuration string which is transmitted to the cell.

Identifiers and commands for the PLU: For correct operation of the PLU, it must be able to recognize only two command strings, namely:

- END

This is a command that sets the PLU in a state in which it waits for the arrival of events from cells (Figure 2).

- DISPATCH (entry number, address)

The PLU enters the value of the address parameter into the address, which is given by the entry number parameter, of the jump table.

In addition, the PLU can recognize an entry as a blank entry. This is accomplished by defining a certain bit pattern as a blank identifier which can be recognized by the PLU.

The jump table: There is a jump table (0506) in the configuration memory. The size of the jump table is

selected, for example, so that there is exactly one single entry for each cell that can be addressed by the PLU. For each cell address there is exactly one single entry in the jump table which can be calculated by the PLU (Figures 5 and 6).

There is a memory address (0601) in an entry in the jump table. This memory address indicates where additional configuration data (0508) from the configuration memory are to be read if there is a feedback from this cell to the PLU.

Start of the system: By resetting the system, the PLU begins to receive or load configuration data from a memory into the configuration memory (0101). All the cells of the unit are in the state in which they can be configured. Then, by loading the program counter (PC) (0505), the PLU jumps to a memory site containing (0102) the address of a start configuration (0507). This start configuration is processed until the PLU recognizes an END identifier (0103). This start configuration programs the unit in such a way that processing of data can begin. After entering the start configuration, the PLU changes on the basis of the END identifier to a state in which it waits for results from the cells (0104).

Arrival of an event from a cell: After processing data, a cell can send a feedback to the PLU. This feedback (event) indicates that the cell and thus the macro in which the cell is contained has completed its work and reloading can begin.

However, before beginning with the loading of a new configuration, the FIFO memory (first-in-first-out memory) described below is processed (0201).

It is advantageous for the memory to be organized as a

FIFO memory. This organization guarantees, for example, that cells which could not be reloaded in the first attempt are guaranteed to be the first in line in the second attempt. This prevents cells which have signaled in the meantime that they can be reconfigured from slipping to the back in processing. In this case, a deadlock situation could occur in which one macro can be reconfigured only when another macro has been reconfigured.

Through the feedback to the PLU, the PLU also receives the address or the number of the cell that triggered the feedback. With this number, the proper entry in the jump table is selected (0203, 0204). The address contained in this entry indicates the start of the configuration to be loaded within the configuration memory (0205).

FIFO memory: The method in accordance with the exemplary embodiment of the present invention takes into account the fact that some cells might not have completed their work, although these cells should already be reloaded. All configuration data of cells in which such a condition applies are copied to a special memory area (FIFO memory) (0506).

Each time before a new configuration is to be loaded, the FIFO memory is run through. Since a new configuration is to be loaded, some cells have completed their work and have entered the "reconfigurable" state. These cells may also include those in which reconfiguration by the PLU has failed in a previous attempt because these cells had not yet completed their work but now this reconfiguration can be performed successfully.

This PLU loads the PC with the contents of the register which indicates the start of FIFO memory (FIFO start REG) (0502) and reads data out of the FIFO memory. A

comparison ascertains whether the end of the FIFO memory has been reached (0301). If this is the case, the system returns to the point in the state machine where reconfiguration is continued (0202).

The FIFO memory is processed like a configuration within the configuration memory. The case can occur where a cell cannot be reconfigured even with another attempt. In this case the configuration data is copied (0302) to this memory location if there is an empty memory location closer to the front of FIFO memory.

This copying operation is accomplished by virtue of the fact that the PLU has stored the start address of the FIFO memory in FIFO start REG (0502) and the end address in FIFO end REG (0503). In addition, the PLU identifies the address of the next free entry (starting from the beginning of the FIFO memory) by means of FIFO free entry REG (0504, 0303). After the configuration string has been copied (0304) to the free entry, the PLU positions the pointer of FIFO free entry REG at the next free entry (0305) within the FIFO memory. The search is then conducted in the direction of the end of the FIFO memory. Then the PC is set at the next entry within the FIFO memory (0306).

Reloading cells: The PLU then reads the configuration data out of the configuration memory. This data contains the address of the cell to be reloaded (Figure 4). Each cell can signal that it can be reloaded. The PLU tests this (0401). If the cell can be reloaded, the configuration data is transferred from the PLU to the cell.

If the cell is not yet ready, the data read by the PLU are written to a memory area, the FIFO memory, within the configuration memory (0402). The address to which the

data is written is stored in a register (FIFO end REG) (0503) in the PLU.

This process is repeated until the PLU recognizes the END identifier of the configuration program and returns it to the state in which the PLU waits for events from the cells (0403).

Structure of the configuration program: After a cell has given the signal for reloading and the macro in which the cell is integrated has been reloaded, a new configuration is obtained. The cell which has previously given the signal to the PLU can now have a very different function, in particular it may no longer be the cell which sends a reload signal to the PLU. In the new configuration, it is possible for the same cell to again send the reload signal to the PLU.

By means of the DISPATCH command within the configuration program, a new address can be written to the entry position of the cell in the jump table (0604). This new address may point to a new configuration or subconfiguration to be loaded upon feedback from this cell.

Figure 1 shows a flow chart of the steps to be carried out after a system start. The system goes to the waiting state (0104) upon comparison of the start configuration with the END identifier.

Figure 2 shows a flow chart of the required steps to be carried out during the waiting state and after a reconfiguration has been signaled by a cell. The flow chart has an entry point (0202) which is accessed from another location.

Figure 3 shows a flow chart of how the FIFO memory is to

be handled. It also shows how the copy process works within the FIFO memory.

Figure 4 shows in a flow chart which steps are necessary in reconfigurating the cells and how a configuration is processed within the configuration program.

Figure 5 shows the PLU and its registers. The PLU has five different registers, namely:

- The start configuration REG (0501). This register contains the address of the start configuration within the configuration memory. The data is contained in the configuration program in such a way that it can be recognized by the PLU and transferred to the start configuration REG.

- A FIFO start REG (0502). The FIFO start REG indicates the start of the FIFO memory area within the configuration memory.

- A FIFO end REG (0503). The FIFO end REG denotes the end of the FIFO memory. The configuration strings which could not be processed by the PLU are copied to this location.

- A FIFO free entry REG (0504). The FIFO free entry REG indicates the free entry closest to the beginning (FIFO start REG) of the FIFO memory. The configuration strings which again could not be processed by the PLU during the run-through of The FIFO memory are copied to this location.

- A program counter (PC). The PC points to the address within the configuration memory where the next configuration string to be processed by the PLU is located.

- An address REG (0510). The address of a cell to be addressed is stored in this register.

- A data REG (0511). This register stores the configuration data to be sent to the cell addressed by address REG.

- A dispatch REG (0512). Dispatch REG stores the address of the entry in the jump table accessed by the PLU.

In addition, the configuration memory and its various sections are also shown. These are:

- The jump table (0506). For each cell that can be configured by the PLU there is a single entry. This entry contains the address which is loaded into the PC when signaled by this cell.

- A start configuration (0507). The start configuration is any configuration loaded into the unit after starting the system.

- Additional configurations (0508). These configurations can be loaded into the unit during system run time. The configurations consist of configuration strings and PLU commands.

- A FIFO memory area (0509). The FIFO memory area contains all configuration strings that could not be processed successfully in a first attempt.

Figure 6 shows two sections of a configuration. These sections show the commands and configuration strings processed by the PLU. It also shows two sections from the jump table (0601 and 0607) and the status of these sections (0602 and 0608) after processing of the two

configuration sections.

Embodiments: It is assumed that one or more units are to be reconfigured by a PLU as described in the invention. In addition, it is assumed that the system has already loaded the start configuration and that the PLU is in the state of "waiting for an event." The execution begins with the arrival of an event from cell number 41.

The PLU begins first with the processing of The FIFO memory (0201). The start of the FIFO memory is transferred to the PC from the FIFO start REG register. The data at the location to which the PC is pointing is read. Then a check is performed to determine whether the end of the FIFO memory has been reached. This is the case in this embodiment, because the system is being reloaded for the first time.

The address of the cell which has triggered the signal is converted by the PLU to an address in the jump table. This calculated address is loaded into the dispatch REG (0512). The PLU then reads the address out of the jump table (0506) which is stored at the memory address addressed by the dispatch REG (0601). This address is loaded into the PC.

Then the processing of the configuration strings begins (0603). It is assumed that command number 3 (1.3 MUL) cannot be executed because the cell with the address (1.3) cannot be reconfigured. The data is then copied to the FIFO memory. On reaching the DISPATCH command (0604), a new address is entered (0602) at address 41 in the jump table. The END command again puts the PLU in the state of "waiting for an event."

After a period of time, a signal again arrives from cell 41. Now there is another address (0602) at address 42 in

the jump table. The PLU again processes the FIFO memory first. Now the data is in the FIFO memory.

The data from the FIFO memory is read and an attempt is made to load the addressed cell with the data. Since the cell can now be reconfigured, this attempt is successful. A blank identifier is then written into the entry in the FIFO memory .

The original processing is continued and the reading of configuration data then begins at a different address (0605).

This configuration is processed; this time the DISPATCH command writes an address into entry number 12 of the jump table (0606). Then the END command causes the PLU to return to the state of "waiting for an event."

This interplay is repeated for the entire run time of the system.

Definition of terms

Configurable element: A configurable element is a unit of a logic unit which can be set by a configuration string for a specific function. Configurable elements are thus all types of RAM cells, multiplexers, arithmetic logic units, registers and all types of internal and external interconnection description, etc.

Configuring: Setting the function and interconnection of a configurable element.

Configuration memory: The configuration memory contains one or more configuration strings.

Configuration string: A configuration string consists of a bit sequence of any length. This bit sequence

represents a valid setting for the element to be configured, so that a functional element results.

PLU: Unit for configuring and reconfiguring programmable units. Embodied by a state machine or a microcontroller adapted specifically to its function.

Macro: A macro is a quantity of cells which together implement a task, function, etc.

Reconfiguring: New configuration of any number of configurable elements of a programmable unit while any remaining number of configurable elements continue their own functions (see configuring).

Feedback: Feedback is an action that can be triggered by a cell. With feedback, various pieces of information can be sent to the unit receiving the feedback.

Cell: See configurable element.

State machine: Logic unit which can assume various states. The transitions between states depend on various input parameters. These are known machines for controlling complex functions.